Decision Making in a Robotic Architecture for Autonomy

Tara Estlin, Rich Volpe, Issa Nesnas, Darren Mutz, Forest Fisher, Barbara Engelhardt, and Steve Chien

Jet Propulsion Laboratory

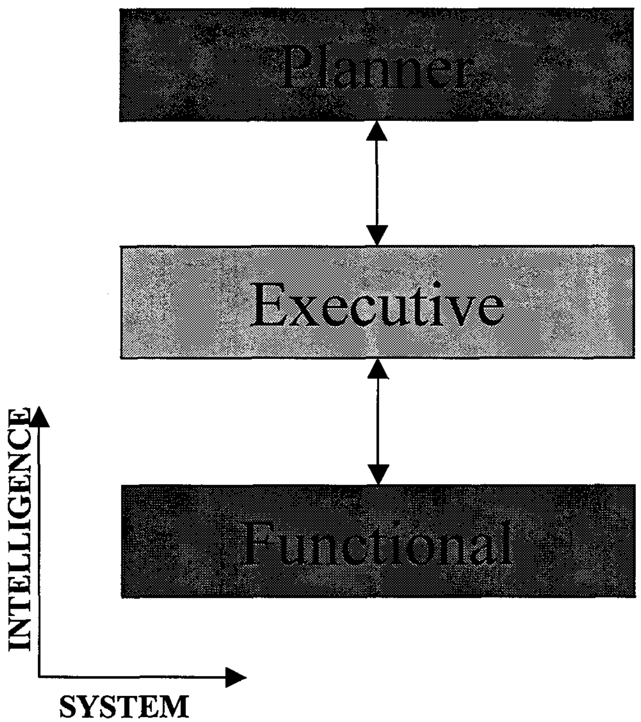
California Institute of Technology

Presented by Rich Volpe

Background

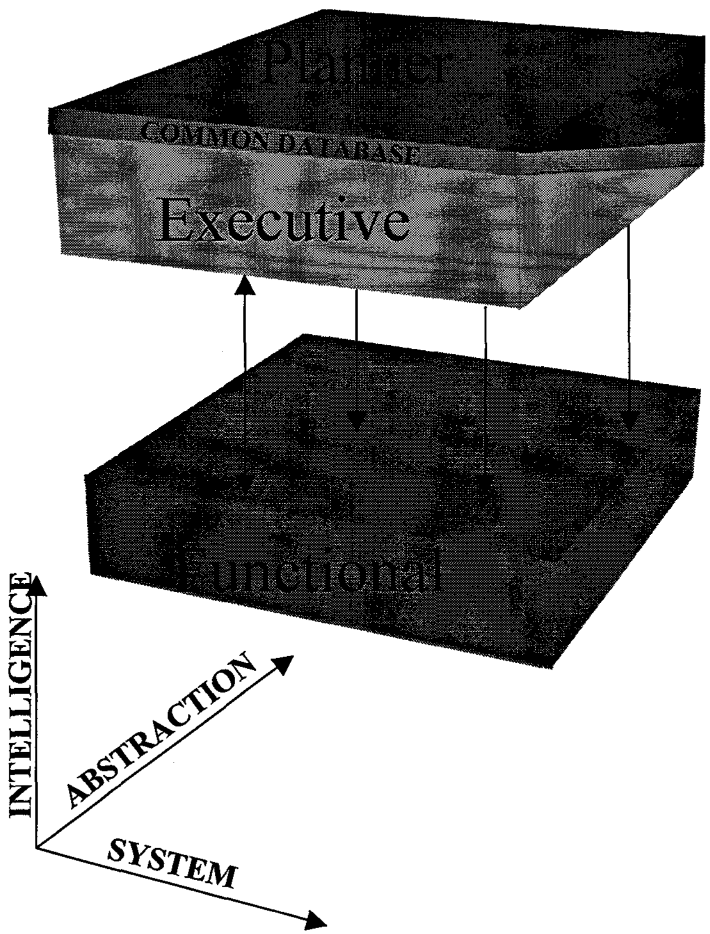
- CLARAty is an architecture for robotic control being developed at JPL
- CLARAty being designed to address future mission autonomy concerns
  - Many upcoming rover missions
  - Rovers will travel significantly farther distances requiring more autonomous capabilities
- Top layer of CLARAty provides autonomous decision-making and rover-sequencing capabilities (Primary focus of talk)
- Bottom layer provides basic robot functionality

Review of 3 Layer Approach



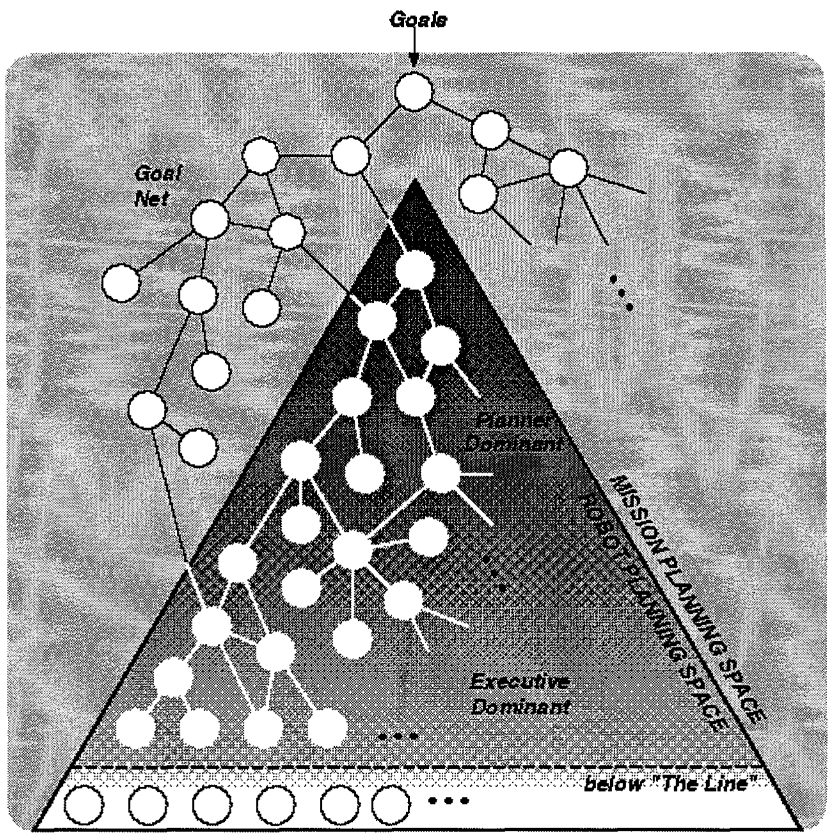
- Most past robotic architectures comprised of three layers
- Layers are organized by level of abstraction and intelligence in which they operate
- Several significant limitations
  - Each layer has separate representation and model of robot and environment
  - Creates unnecessary overhead to translate between layers and maintain separate models
  - Each layer also constrained to work at certain level of plan abstraction
  - Does not allow for use of new techniques which closely integrate planner and executive functionality

CLARAty Two-Layer Architecture



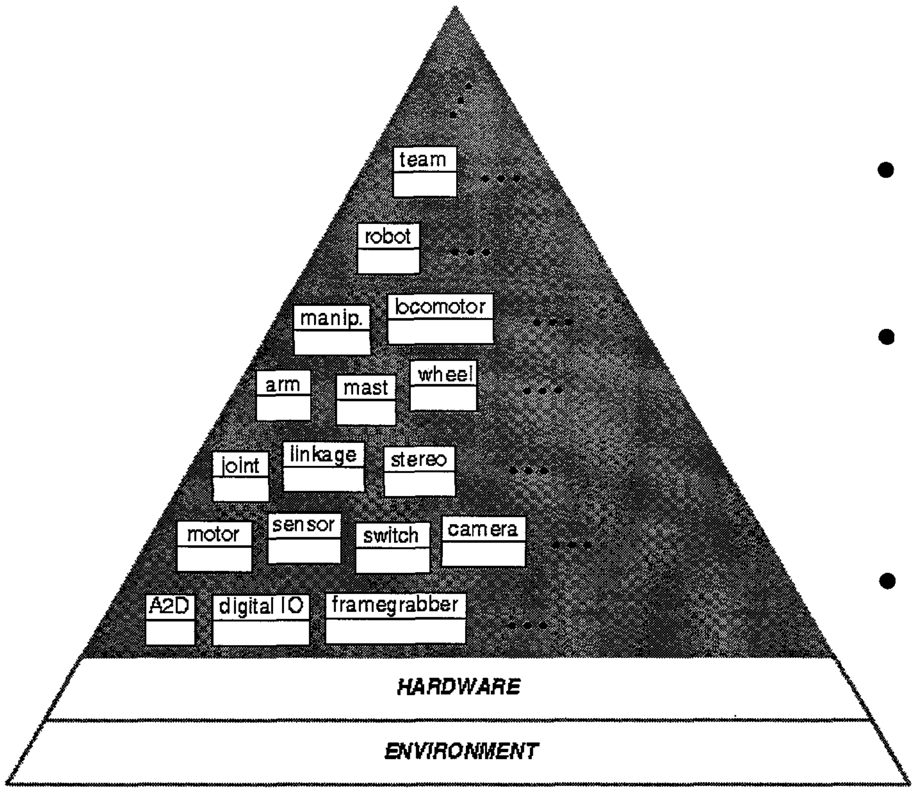
- Provides evolution to two-tiered architecture
- Each tier operates at all levels of abstraction
- Blends declarative and procedural techniques for decision-making
- Planner and exec can be closely integrated and even use common database
- Planning tier can access state and resource information contained in Functional Layer
- Interface between layers is flexible and allows user or domain to dictate capabilities used

Intro to CLARAty Decision Layer



- Breaks down high-level goals into a plan of activities
- Represents relevant mission and rover constraints (e.g., flight rules, resource limitations)
- Darker portion shows planner domain; lighter portion shows executive domain
- "The Line" separates DL from FL but can be moved
- DL interfaces to FL for commanding, monitoring and resource usage predictions

Intro to CLARAty Functional Layer



- Object-oriented design; FL can be structured to match modularity of hardware
- All objects contain basic functionality for themselves
- State of system is contained in objects and available through query
- Objects also contain predictive knowledge for relevant states and resources
- Designed to interface with rover simulators as well as actual hardware

Decision Layer Implementation

- Utilizes CLEaR (Closed-Loop Error and Recovery) System (Fisher, 2001)
  - Integrates planning and executive techniques
- CLEaR combines CASPER continuous planner (Chien, 2001) and TDL executive (Simmons, 1998)
  - CASPER provides capability for plan generation, limited execution, and re-planning
  - TDL provides full spectrum of executive-type capabilities (e.g., procedural expansion, activity execution and monitoring, exception handling, etc.)
  - CLEaR system intended to provide tightly-coupled approach to coordinating goal-driven and event-driven behavior

CASPER Summary

- CASPER: Continuous Activity Scheduling, Planning, Execution and Re-planning
- Provides continuous planning
  - Generates activity sequence that satisfies set of goals while obeying rover operation and resource constraints
  - Also monitors current rover state and plan execution status
  - Never allows plan to get very far out of sync
  - Utilizes "iterative repair" to repair conflicts that arise in plan by performing plan modifications
  - Can quickly re-plan if new conflicts or opportunities arise
- CASPER has been demonstrated in a number of robotic domains
  - e.g., landed ops for ST4 mission, control of DSN antenna ground station, distributed coordination of multiple rovers

TDL Summary

- TDL: Task Description Language
- Designed to perform robotic task-level control and/or to mediate between planner and rover control s/w
  - Expands tasks into low-level commands
  - Executes commands and monitors execution
  - Handles exceptions
  - Handles task synchronization
- Utilizes "task-tree" structure to describe how tasks (or activities) are broken down into commands
- Successful demonstrated on a number of indoor and outdoor robots
  - e.g., Nomad robot in Antarctica, RWI robot for Mars autonomy navigation

Current CLEaR Framework

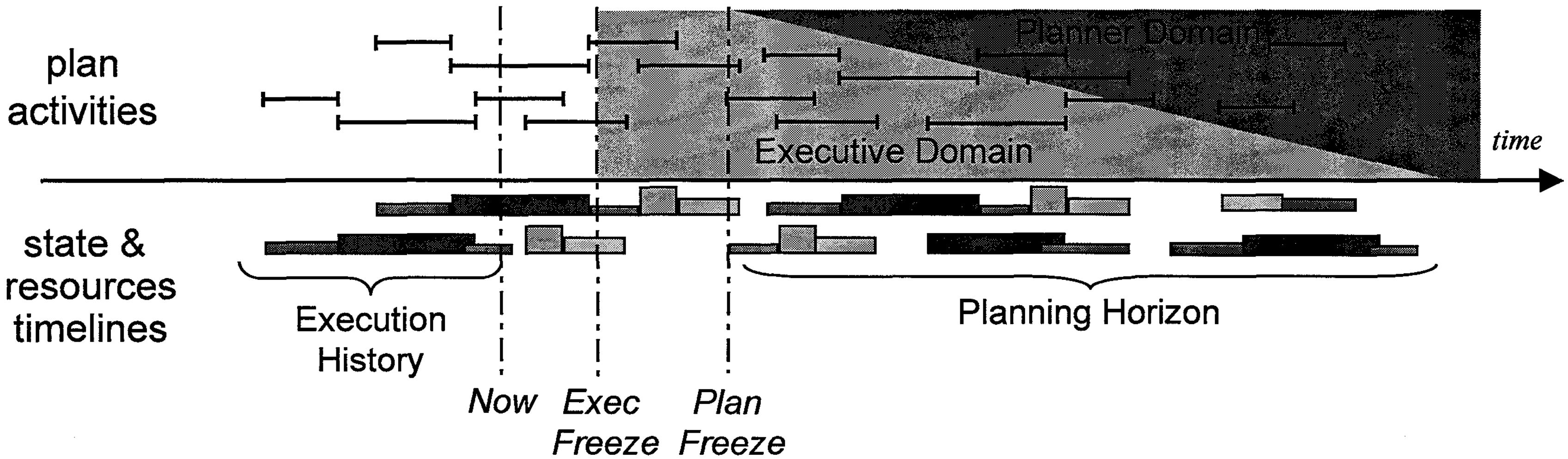
- CASPER and TDL loosely integrated
  - Systems maintain separate plan representations
  - Changes in one plan database can be reflected in other (e.g., TDL modifications can be seen by CASPER and vice versa)
- Decision process for deciding what system should handle conflict and how to coordinate plan changes
  - Can write heuristics to address certain decision types
  - For example,
    - Rover has to avoid unexpected obstacle and gets off-track of planned path
    - Planner and/or executive may react
    - May choose only one reaction or coordinate reactions from both systems

Future CLEaR Framework

- First step, tighten integration of CASPER and TDL
  - Enable CASPER to access procedural capabilities of TDL for initial plan generation and re-planning
  - Increase TDL's knowledge of plan resource levels and limitations; enable this knowledge to influence TDL decisions

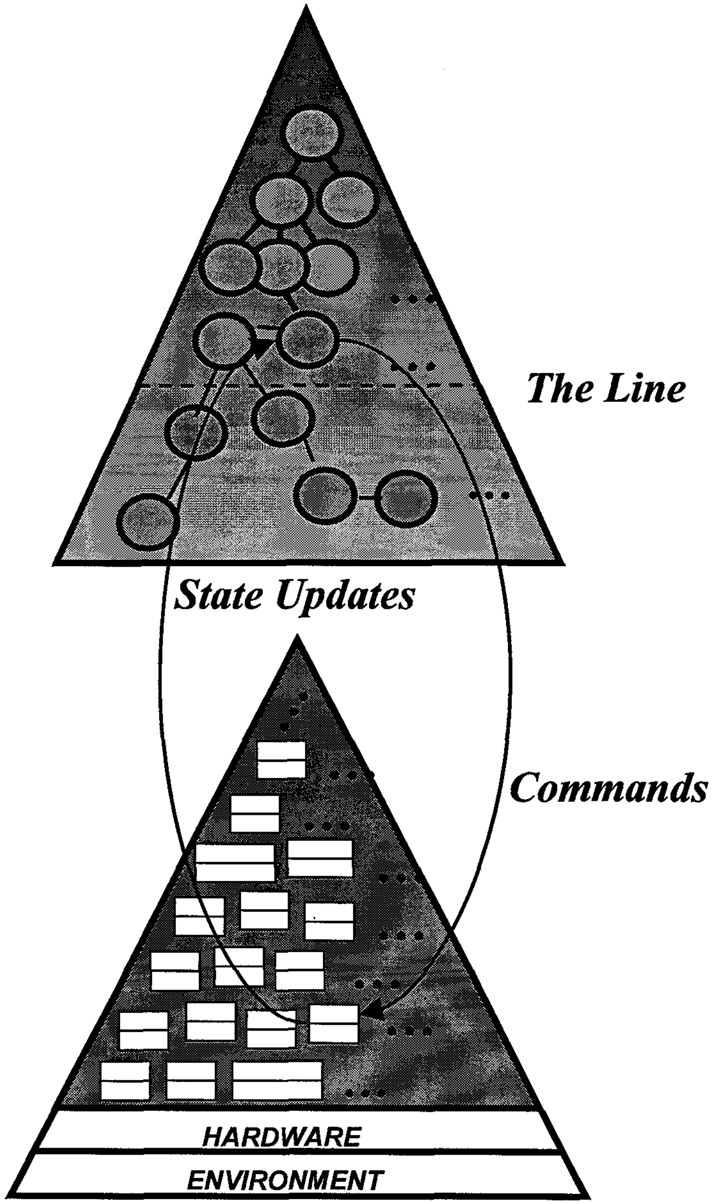
- Finally, fully integrate two systems
  - CASPER and TDL use shared plan representation and plan database
  - Only one domain model utilized by both systems
  - Enable both planning and executive functionality to be used at all levels of activity granularity

Final Planner and Executive Domains



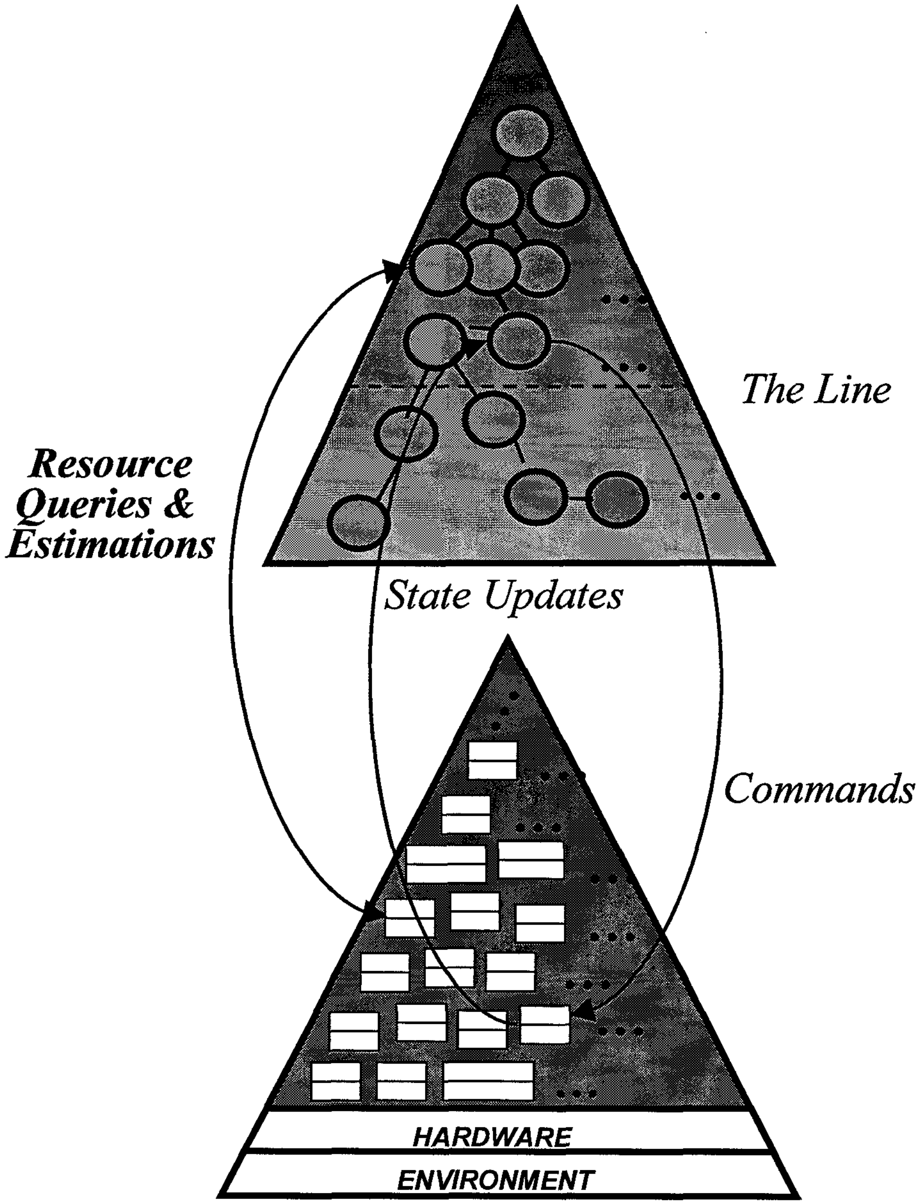
- No strict separation of planner and exec operation domains
- Procedural and declarative capabilities are allowed on both near-term and far-term activities
- Enables procedural constructs (e.g., conditionals, loops) to represent far-term activities
- Enables global-resource analysis to affect near-term decisions

DL Interface to FL: Floating Line



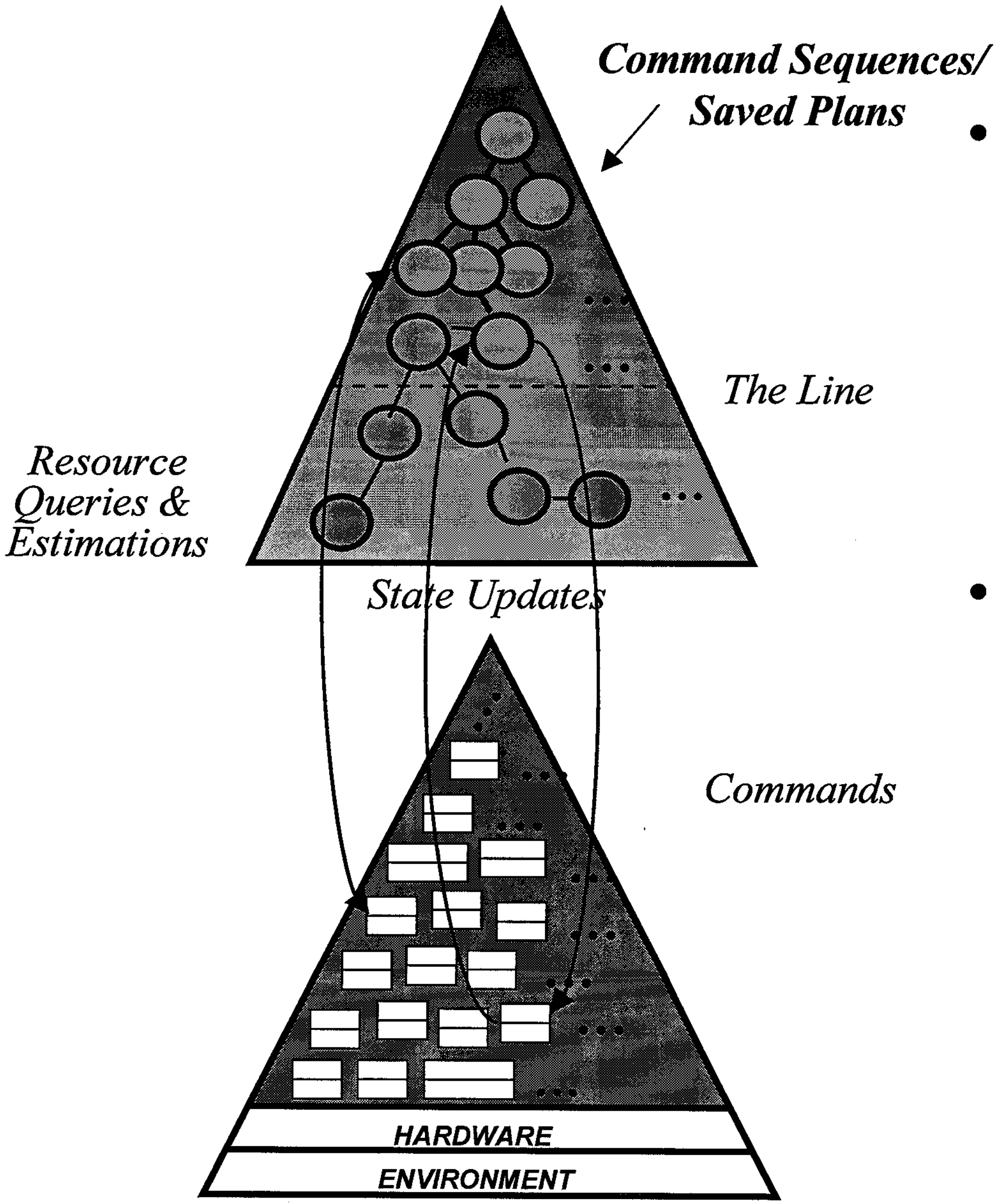
- Activities on fringe of plan network are dispatched to FL
- Status updates (for activities, states and resources) are received from FL
- Elaboration in DL can be to different levels of granularity
- Allowing a "floating line" enables level of control given to DL or FL to be flexible
- Line position can be user or application driven

DL Interface to FL: Resource Querying



- Information on resource usage is kept in FL
- DL can query FL for resource usage predictions
- Queries can be detailed or at a cursory level
- More detailed queries may requires more computational resources
- Queries can contain plan activity parameters and/or other relevant information

DL Interface to FL: Direct Commanding



- Command sequences can be submitted directly from ground ops
  - Simply executed by DL
  - No or minimal modification by planner and/or exec
- Plans or sub-plans can be saved and recalled at future times
  - Plans can be modified by planner to reflect current goals and state of system
  - State updates during execution may cause further modification

Related Work

- Remote Agent Experiment (RAX) (Jonsson, 2001)
  - Flown on NASA Deep Space One (DS1) mission
  - Utilized batch planner and executive to produce and execute command sequences
- Contingent Rover Language (CRL) (Bresina, 1999)
  - Allows both temporal flexibility and contingency branches in rover sequences
  - Produced by Contingent Planner (CPS) and then interpreted by Exec
  - Planner is ground-based; exec is onboard rover
- Other three-layer approaches (Bonasso, 1997; Gat, 1998)
- CPEF (Myers, 1998)
  - Related approach to combining planning and execution
  - Incorporates significant user guidance module (more ground-based)
  - Demonstrated for air-campaign plans

Conclusions

- CLARAty is a robotic architecture that provides autonomous rover control
- The CLARAty Decision Layer provides intelligent decision-making capabilities
  - Supports closely integrated planning and executive techniques
  - Produces and continually refines command sequences
  - Interfaces with Functional Layer in flexible fashion to provides different levels of DL capabilities
- CLARAty is currently being applied to several robotic efforts at JPL and is being directed towards future flight implementations